دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر

کارشناسی ارشد علوم کامپیوتر گرایش داده کاوی

پروژه شماره ۷ درس داده کاوی

نگارش

حدیث حق‌شناس جزی

استاد راهنما

دکتر مهدی قطعی

استاد مشاور

آقای بهنام یوسفی مهر

دی ۱۴۰۱

چکیده

در این گزارش هدف این است که با استفاده از دیتای creditcard که شامل داده هایی از کارت های اعتباری است ، ابتدا به پیش پردازش داده که شامل حذف داده های تکراری سطری و بررسی همواستگی بین ویژگی ها و مواردی از این قبیل بپردازیم سپس به کمک متد under sampling که روشی برای متوازن سازی داده ها از طریق نگه داری داده های تقلبی و حذف داده های غیر تقلبی است، اقدام به یکنواخت سازی دیتافریم جهت آموزش بهتر در قسمت مجموعه آموزشی کرده و سپس به کمک روش pca داده ها را در ۲ بعد کاهش میدهیم تا بتوانیم از متد های خوشه بندی متفاوت برای شناخت و بررسی داده های تقلبی نظیر dbscan، درخت تصمیم و ... استفاده کنیم .

فهرست مطالب

فصل اول

مقدمه

مقدمه

مهمترین بخش در شناسایی تراکنش های تقلبی، شناخت و پیش پردازش دیتافریم است. که در ادامه به آن خواهیم پرداخت. این داده که شامل ۳۱ ستون میباشد، ۲۸ ستون با نام های غیر مشخص دارد. یعنی ما اطلاعی از اینکه هر ستون بیانگر چه چیزی است در دست نداریم. در ستون class نوع تراکنش (تقلبی یا غیر تقلبی بودن) مشخص شده است و ۲ ستون زمان و مقدار تراکنش نیز موجود میباشد.

فصل دوم

پیش پردازش داده ها

پیش پردازش دیتاست

در ابتدا فایل دیتا را فراخوانی میکنیم. اطلاعات کلی در تصویرهای زیر قابل مشاهده است:

```
df= pd.read_csv("creditcard.csv")
```

```
df.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V2
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.12853
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.16717
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.32764
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.64737
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.20601

5 rows × 31 columns

```
df.shape
```

```
(284807, 31)
```

```
df.describe()
```

V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
70e+05	...	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	284807.000000	284807.000000
55e-15	...	1.656562e-16	-3.444850e-16	2.578648e-16	4.471968e-15	5.340915e-16	1.687098e-15	-3.666453e-16	-1.220404e-16	88.349619	0.001727
32e+00	...	7.345240e-01	7.257016e-01	6.244603e-01	6.056471e-01	5.212781e-01	4.822270e-01	4.036325e-01	3.300833e-01	250.120109	0.041527
07e+01	...	-3.483038e+01	-1.093314e+01	-4.480774e+01	-2.836627e+00	-1.029540e+01	-2.604551e+00	-2.256568e+01	-1.543008e+01	0.000000	0.000000
76e-01	...	-2.283949e-01	-5.423504e-01	-1.618463e-01	-3.545861e-01	-3.171451e-01	-3.269839e-01	-7.083953e-02	-5.295979e-02	5.600000	0.000000
73e-02	...	-2.945017e-02	6.781943e-03	-1.119293e-02	4.097606e-02	1.659350e-02	-5.213911e-02	1.342146e-03	1.124383e-02	22.000000	0.000000
90e-01	...	1.863772e-01	5.285536e-01	1.476421e-01	4.395266e-01	3.507156e-01	2.409522e-01	9.104512e-02	7.827995e-02	77.165000	0.000000
99e+01	...	2.720284e+01	1.050309e+01	2.252841e+01	4.584549e+00	7.519589e+00	3.517346e+00	3.161220e+01	3.384781e+01	25691.160000	1.000000

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

با دستور describe اطلاعات آماری و چارک داده هارا مشاهده میکنیم. با توجه به میانگین های متفاوت داده ها، پر واضح است که داده در ستون زمان و مقدار تراکنش ها نرمال نمیباشد که در ادامه اقدام به نرمال سازی میکنیم.

پر کردن داده های خالی

سپس به بررسی داده های خالی یا missing data میپردازیم. همانطور که مشاهده میشود داده های خالی نداریم اما در سطر های داده تکراری موجود است.

```
1 df[df.duplicated()]
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24
33	26.0	-0.529912	0.873892	1.347247	0.145457	0.414209	0.100223	0.711206	0.176066	-0.286717	...	0.046949	0.208105	-0.185548	0.001031
35	26.0	-0.535388	0.865268	1.351076	0.147575	0.433680	0.086983	0.693039	0.179742	-0.285642	...	0.049526	0.206537	-0.187108	0.000753
113	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089	0.023092	-0.626463
114	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089	0.023092	-0.626463
115	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089	0.023092	-0.626463
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
282987	171288.0	1.912550	-0.455240	-1.750654	0.454324	2.089130	4.160019	-0.881302	1.081750	1.022928	...	-0.524067	-1.337510	0.473943	0.616683
283483	171627.0	-1.464380	1.368119	0.815992	-0.601282	-0.689115	-0.487154	-0.303778	0.884953	0.054065	...	0.287217	0.947825	-0.218773	0.082926
283485	171627.0	-1.457978	1.378203	0.811515	-0.603760	-0.711883	-0.471672	-0.282535	0.880654	0.052808	...	0.284205	0.949659	-0.216949	0.083250
284191	172233.0	-2.667936	3.160505	-3.355984	1.007845	-0.377397	-0.109730	-0.667233	2.309700	-1.639306	...	0.391483	0.266536	-0.079853	-0.096395
284193	172233.0	-2.691642	3.123168	-3.339407	1.017018	-0.293095	-0.167054	-0.745886	2.325616	-1.634651	...	0.402639	0.259746	-0.086606	-0.097597

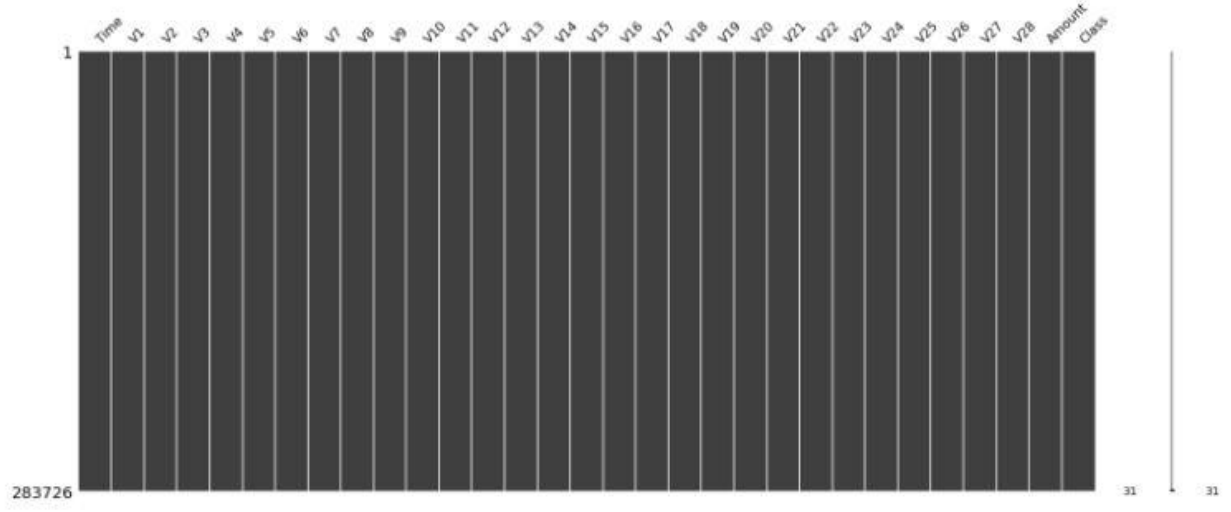
1081 rows × 31 columns

```
1 df= df.drop_duplicates()
```

```
1 df.shape
```

```
(283726, 31)
```

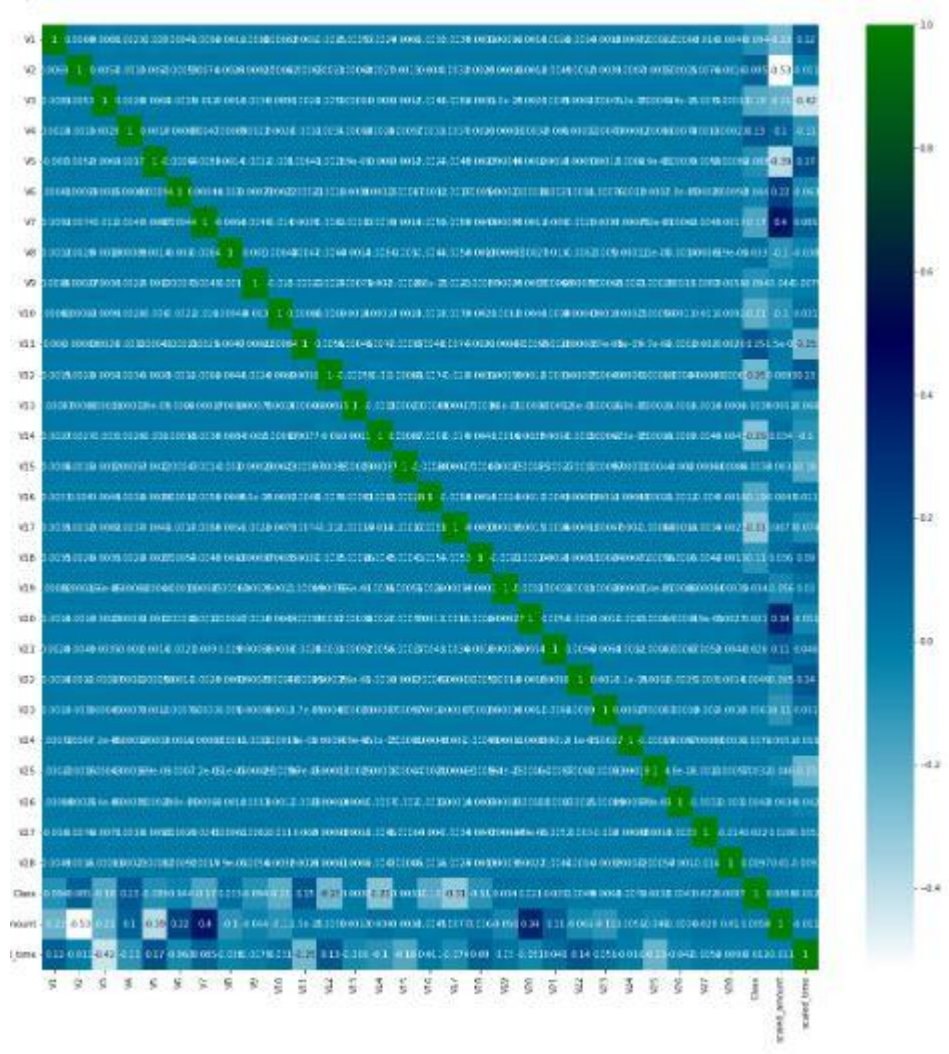
```
1 msno.matrix(df)
2 plt.figure()
3 plt.show()
```




```
<Figure size 432x288 with 0 Axes>
```

بررسی همبستگی ویژگی ها

حالا به کمک ماتریس همبستگی نمودار زیر را رسم میکنیم. در این داده ستونی که نیاز به حذف باشد وجود ندارد. پس به بخش بعد میرویم.



.

بررسی داده های تقلبی و نرمال سازی :

در این بخش به بررسی میزان داده های تقلبی و مصور سازی آنها پرداخته ایم سپس به کمک under sampling داده را متوازن سازی کرده ایم که در تصاویر زیر مراحل قابل مشاهده است:

همچنین در این بخش به نرمال سازی ۲ ستون مقدار و زمان نیز پیداخته ایم.

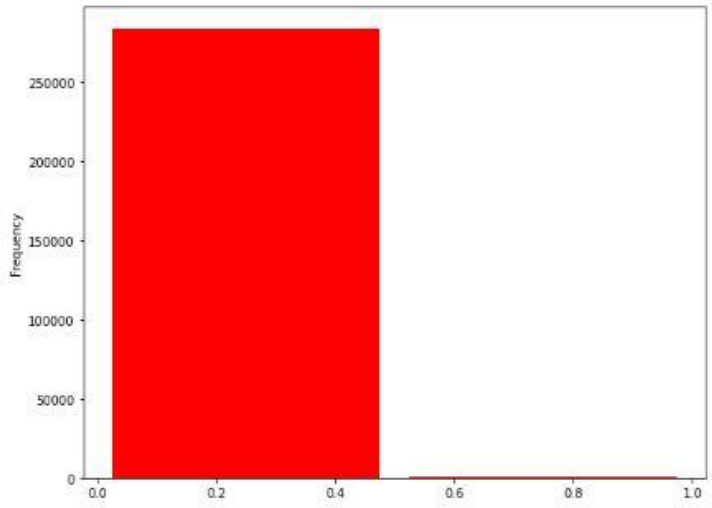
```
len_class= len(df['Class'])
print(df['Class'].value_counts()[0])
((df['Class'].value_counts()[0])/len(df))*100
```

```
283253

99.83328986416473
```

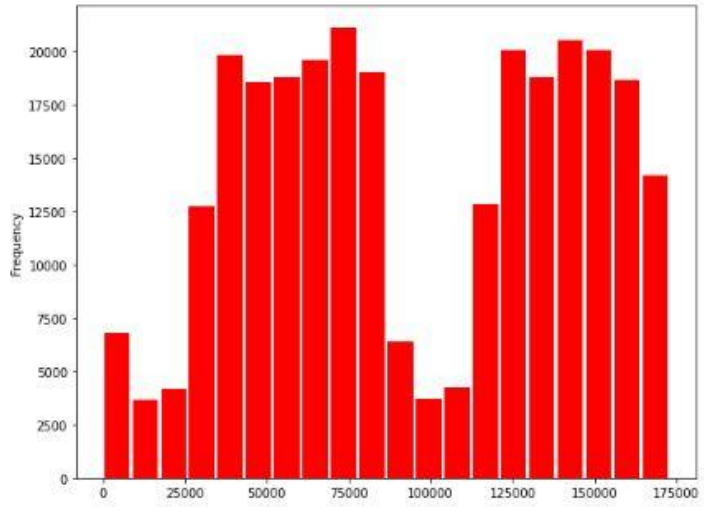
```
print(df['Class'].value_counts()[1])
((df['Class'].value_counts()[1])/len_class)*100
```

```
473

0.1667101358352777
```

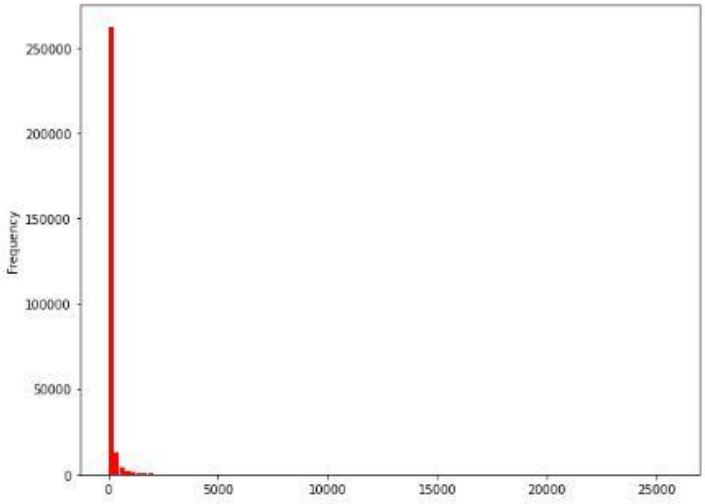
```
plt.figure(figsize=(9,7))
df['Class'].plot(kind = 'hist', bins= 2 , rwidth=0.9, color="r");
```



```
plt.figure(figsize=(9,7))
df['Time'].plot(kind = 'hist', bins= 20 , rwidth=0.9, color="r");
```



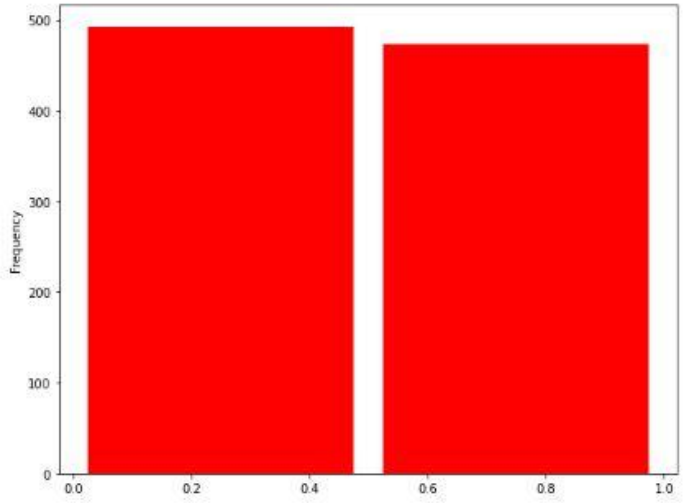

```
plt.figure(figsize=(9,7))
df['Amount'].plot(kind = 'hist', bins= 100 , rwidth=0.9, color= "r");
```



```
df.head()
```

V6	V7	V8	V9	V10	...	V22	V23	V24	V25	V26	V27	V28	Class	scaled_amount	scaled_time
388	0.239599	0.098698	0.363787	0.090794	...	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	0	1.774718	-0.995290
361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	0	-0.268530	-0.995290
499	0.791461	0.247676	-1.514654	0.207643	...	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	0	4.959811	-0.995279
203	0.237609	0.377436	-1.387024	-0.054952	...	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	0	1.411487	-0.995279
921	0.592941	-0.270533	0.817739	0.753074	...	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	0	0.667362	-0.995267

```
plt.figure(figsize=(9,7))
new_df['Class'].plot(kind = 'hist', bins= 2 , rwidth=0.9, color="r");
```



کاهش ابعاد :

```
X = new_df.drop('Class', axis=1)
y = new_df['Class']

from sklearn.decomposition import PCA
X_reduced_pca = PCA(n_components=2, random_state=0).fit_transform(X.values)
X_reduced_pca
```

```
array([[-9.11279833,  1.30081699],
       [ 2.9991364 , -3.63165599],
       [18.40971152, -2.61363766],
       ...,
       [-8.88645361,  1.24946201],
       [18.1524617 , -3.47515368],
       [ 0.05653473, -4.27847072]])
```

```
from sklearn.cluster import DBSCAN
from sklearn import metrics

from sklearn.metrics import davies_bouldin_score, silhouette_score, calinski_harabasz_score


dbscan = DBSCAN(eps = 1.4 , min_samples = 8)
y_dbscan = dbscan.fit_predict(X_reduced_pca)

print("davies_bouldin_score is: ")
print(round(davies_bouldin_score(X_reduced_pca, y_dbscan), 3))
print("silhouette_score is: ")
print(round(silhouette_score(X_reduced_pca, y_dbscan), 3))
print("calinski_harabasz_score is: ")
print(round(calinski_harabasz_score(X_reduced_pca, y_dbscan), 3))

```

```
davies_bouldin_score is:
1.082
silhouette_score is:
0.624
calinski_harabasz_score is:
686.231
```

منابع و مراجع:

- سایت کگل

پایان